BLACK LOWE & GRAHAM
816 2nd Ave., Seattle, WA 98104
206-381-3300 [HERO-1-1111]

# APPENDIX A

BLACK LOWE & GRAHAM
816 2nd Ave., Seattle, WA 98104
206-381-3300 [HERO-1-1111]

# FlexCube 2000 Prototype Requirements and Implementation Notes

## July 16, 1999

# Table of Contents

# Executive Summary

## Introduction

The FlexCube System has the potential to meet HHN's current and future patient management requirements. A prototype system is under development, providing a platform for testing and further investigation of this system. See the section on Strategic Requirements for more detailed information about FlexCube.

Ultimately, a production implementation of this system will be different from this initial prototype. The prototype system will fit on a laptop computer and use a Windows GUI for all functionality. A production system would differ in several respects. The patient management portion would run as a thin-client Web-based application. Minor adjustments to the UI would be required in this conversion. The database used for initial testing will not be scalable to handle full production volume. However, the schema should be relatively easy to migrate to a full production database, such as Oracle.

This prototype is not meant to be the final system implementation. It will allow HHN to develop and explore the FlexCube concept. Many of the new software objects will be reusable in later stages of development and hopefully by the core engineering group. The ultimate goal is for this system to radically improve content development and patient care, while reducing the human resources necessary to manage a given population.

## Implementation

The initial prototype will be developed to run on a single laptop computer. A serial cable will connect the laptop to a Health Buddy appliance, enabling round-trip communication. The prototype will only be able to distribute content to remote appliances by tapping into HHN's existing data-center infrastructure. Difficulties are expected in mapping patient response and risk-analysis data between the two systems since they are based on completely different paradigms. However, it should be straightforward to convert FlexCube sessions into traditional surveys to allow patients to test raw output from the system. This conversion will be explored more fully in the near future. Another option is to build a mini-patient server to allow larger scale testing.

Several new "objects" will need to be developed in order to implement the FlexCube system. Cube and dialog objects are at the core of the system. These objects will require a Health Buddy Database (BDB) and enhanced scripting language in the appliance to fully operate. A single schema will be leveraged to act as both remote and data-center database. Ultimately, content will be composed off-line and exchanged as BDBs between the data-center and remote installations.

Desktop Applications will be developed under Windows using Borland C++ Builder 4. This system includes many pre-packaged GUI controls and database support necessary for prototype development. Care will be taken to avoid using non-standard database and GUI techniques, easing migration to a larger system.

Plans will be finalized for the FlexCube prototype architecture, resulting in an Application Server Schema, Windows GUI design, Application Server Requirements, Appliance Application Requirements. The final design should allow application servers to become modules, like a Lego set. For example, each care manager in an organization could have a full application server. A separate DSS server could track content and results for aggregate reporting.

Appliance development will be based on the DemoBox project. Structures and BDBs will be developed and integrated into the scripting language. Later, a library of script sub-routines will utilize these features to support the cube and dialogs required by the FlexCube system.

The NBV DemoBox Composer application will be leveraged into a dialog editor for the FlexCube system.

## Resource Utilization

**Erik Jensen** (Creator of the FlexCube concept) – Will provide overall management, direction in the functional and user-interaction portions of the system.

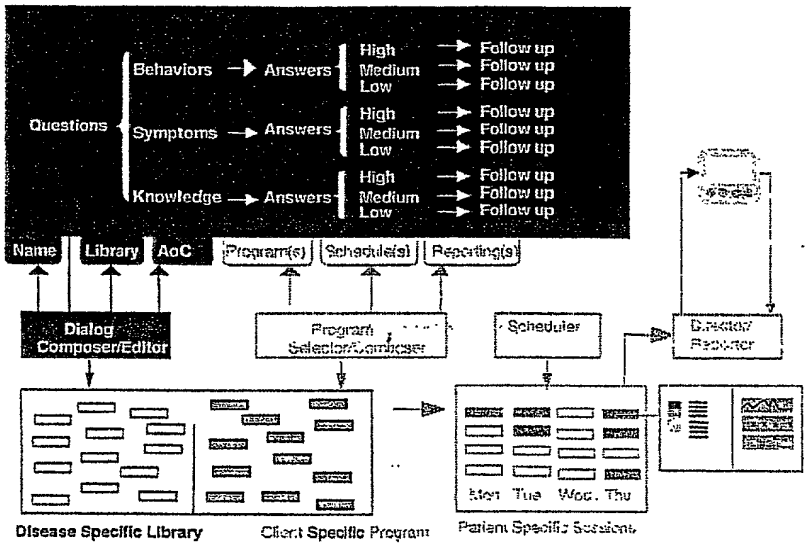
**Daniel Lindsey** – Will provide engineering direction and guidance throughout the production of the prototype system. Work with Albert Bodenhamer to design databases.

**Albert Bodenhamer** – Will be responsible for implementing Windows GUI and database portions of the system.

**Adam Wozniak** – Will be responsible for implementing the enhanced scripting language and support scripts.

**Eric Smith** – Will be responsible for implementing BDBs, VM support, appliance file system, and communications.

## FlexCube




## System Components

- **Health Buddy Database (BDB)** – A micro database object that moves cube content, dialogs, and raw response information to and from the appliance via XML
- **XML Schema Objects** – For transporting Patient Cubes, Dialogs, Box Control Info, etc.
- **Application Server Schema** – Stores patient data, patient demographics, personalized program/dialog content, patient cube-state, and raw response data

## Applications

### *Appliance*

- **Box Manager Application** – High level control over appliance functionality, implements to-do list
- **Session Manager** – Manages To Do list, launches content applications
- **FlexCube Presenter App** – Provides sessions by executing dialogs and updating cube state
- **User Login Manager** – Enables patient login and authentication
- **Communications Manager** – Overall responsibility for non-conversational communications
- **Connection Manager** – Brings up physical connection
- **Login Manager** – Responsible for logging into host system
- **Mail Exchange Manager** – Responsible for data exchange between appliance and host

### *Windows GUI*

- **Administrator Application** – Provides system maintenance and data entry features, account setup, patient entry, etc.
- **Dialog Editor / Composer** – Enables the creation and maintenance of non-personalized dialogs within libraries, dialog labeling, and import/export capability to the application server database
- **Program Selector** – Selects dialogs from libraries in and out of content specific programs, specifies labels for scheduling and reporting
- **Personalizer** – Maintains patient-specific cube metrics, defining risk levels, etc.

- **Session Scheduler** – Semi-automatic creation and scheduling of patient specific sessions
- **Reporter** – Patient results lookup organized into levels from patient-specific program overview to raw results from particular session, including graphical body system view.

## Future Additions

- **Enhanced Scripting Language** – Enables the appliance to manipulate BDBs, supporting raw data-collection and cube / dialog interaction
- **Appliance Apps** – Script Language implementation of appliance applications.
- **Thin-Client Reporter** – Allows system access via the Web
- **Global Patient List Server** – This server maintains a list of all patients that HHN has serviced. As new patients are added, they will be assigned a Globally Unique ID, which will follow them around our service.

# Implementation Strategy

## *Milestone 1 – Build Foundation and Design System*

Initially, construction will begin on foundation objects and code while design decisions are being finalized for the system architecture. Foundation work includes BDB implementation, Enhanced Script Environment, and enhancements leveraging NBV Composer into Dialog Composer. The final system will require these components regardless of database & GUI design. These components will all be integrated into the final release. System architecture, project schedule, GUI design and other project decisions will be completed in parallel.

- Rough-draft GUI design
- Project Schedule
- Scripting Language Selected
- Architectural decisions finalized
- Rough-draft application server schema
- BDB library code, suitable for integration onto appliance and Windows
- Enhanced Scripting environment with HAL support
- Dialog Editor with branching, calculation nodes, decision nodes, etc.

## *Milestone 2 – Present a Session on the Appliance*

By Milestone 2, the appliance will be capable of downloading a session generated by the Windows apps and present the session to a patient.

- Finalized Schema
- Finalized GUI design
- Windows apps scheduling content into sessions
- Windows apps generating output XML
- Appliance reading content XML into internal format
- Presenter application on appliance able to read BDB and present a session

## *Milestone 3 – Round-Trip between Server and Appliance*

By Milestone 3, the system will be able to generate sessions, run them on the appliance, retrieve the results, and display raw and refined results.

- Windows apps sending scheduled session content and receiving results over the serial cable
- Appliance presenting the session and recording results
- Windows apps uploading results from appliance and displaying raw response and cube data

## *Milestone 4 – System fully Alpha*

By Milestone 4, the appliance will be polished and dependable through the serial link. The Windows apps will be fully functional and include all essential functionality to make the system work. Wizards will have been added to support appropriate system functions, especially for content composition

- Windows apps generating reports and displaying patient status
- Wizard functionality in Composer and key parts of Reporter
- Appliance ready to demo

# Strategic Requirements

## *Database Highly Scalable*

We ultimately need to support 10's of thousands of users and Millions of appliances.

## *Portable Content and Results*

Other appliances and systems should be capable of presenting and processing our content

## *Secure and Reliable*

The system must provide controlled access to patient data, system functionality, and maintain secure communications

## *Automatic Generation of Adaptive Content*

The system should focus content where each individual patient requires the most attention. This needs to happen automatically so that we can support very large patient populations with a small group of Care Managers.

## *Multiple Patients per Appliance*

This will open up the possibility of installing appliances at centralized locations, such as pharmacies. This will also enable families to share an appliance within a home.

## *Multiple Content Senders*

Each patient using a Health Buddy needs the ability to subscribe to multiple content providers. For example, a patient from Kaiser might subscribe to a diabetes program, a weightloss program, and a health newsletter from another organization.

## *Multi-Lingual Parallel Content*

The system must support the large number of non-English speakers throughout the US. This feature creates the potential for a global marketplace.

## *Independent Appliance Decisions based on Patient Data*

Immediate Patient Feed-Back
Ability to Score SF36 on Appliance

## *Ad Hoc patient communication*

Care providers need to be able to send personal messages and queries outside of a patient's normal programs.

## *Appealing Graphical UI*

The Graphical User Interface must be pleasant for the care providers who will be using our system for hours at a time.

## *Interface to Patient List Server*

This system needs to communicate with a centralized server that issues Globally Unique Patient Ids. This will facilitate knowing that John Smith at Kaiser is the same John Smith at Stanford's Clinical Research Program, etc.

## *Reports Summarized by Hierarchy Structure*

This will enable the system to mesh with future Information Systems.

# System Components

## Application Server Schema/Database

The Application Server Schema is the heart of the FlexCube system. All data flows in and out of the database. Content development generally occurs offsite on a fat client installation. Content is imported and exported between the installation and the actual data center. Initially, fat client tools will be used for managing patients and reviewing results. In the next phase, thin client tools will enable Internet access via the Web.

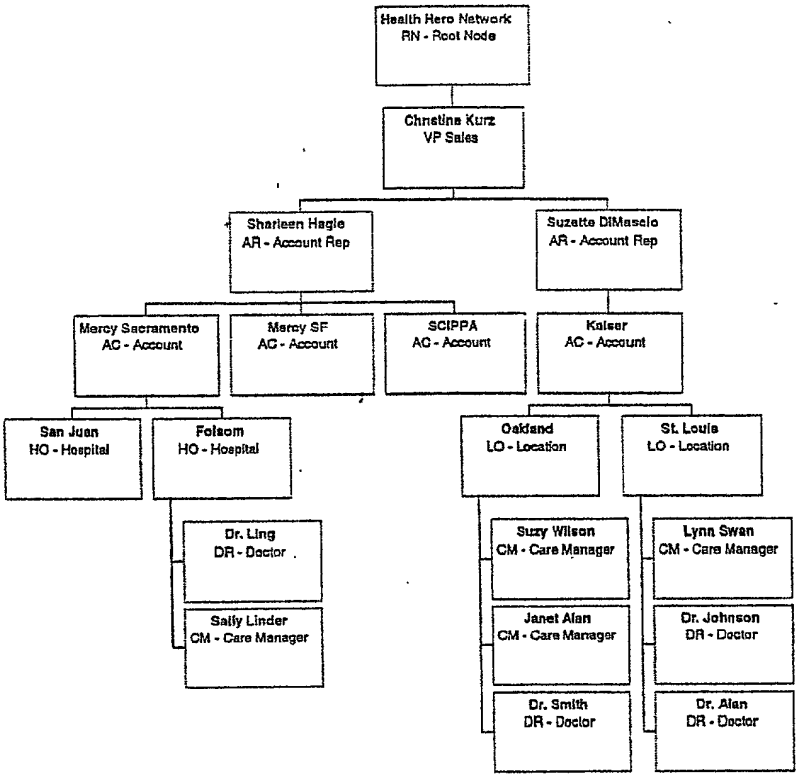
### *Requirements*

- As simple as possible to support scaling – support 10's of thousands of users and millions of appliances
- Fast Cube state retrieval for an arbitrary date/time stamp enabling reports to be generated quickly
- Accounts are always responsible for their own programs
- Ability to Import and Export content between Application Servers
- Ability to aggregate data from multiple Application Servers for generating aggregate DSS reporting
- Support Arbitrary Client Hierarchy
- Support the following Objects
  - Account Entity / Hierarchy – Support arbitrary account hierarchy
    - Group – Allow Care Managers to arbitrarily group their patients
    - Patient Notes
  - Patient
    - Patient Demographics
    - Medications
    - Escalations
  - Patient Cube States
  - Program
  - Dialogs
  - Raw Response Data
  - Program Session Schedules

### *Components / Schema Objects*

- System Table – Include Server GUID
- Account Entity / Hierarchy
- Group
- Patient
  - Demographics
  - Acct Entity Notes
  - Medications
  - Escalations
- Program
- Patient Cube – One Cube per Patient or Multiples for each program?
- Dialog
- Response Data
- Session

# Sample Application Server Hierarchy




**Account Entity / Hierarchy**

Account Entities are like entries on an org chart. The schema supports several different types of entries or nodes. The hierarchy is split into two main sections: HHN and Account. The top portion is reserved for HHN's sales structure and the bottom portion is available for account entries. Much of the structure in both sections will likely be filler. By modeling both HHN's and the account's structure, the system will be able to produce reports summarized by each level up and down the hierarchy.

In order to operate, the system needs several specific types of entries. HHN's Root must be at the top in each Application Server. Logging into the system as root disables all security features and enables system maintenance features. Account Representatives must be at the bottom of HHN's portion. Filler entries allow HHN's real sales structure to be modeled.

Only account entries can be entered under an HHN Account Representative. Programs that are shared within an account will be associated with these account entries. These entries establish the root of each account within the system.

Doctors and Care Managers are always at the bottom of an account's hierarchy. These are considered the users of the system. The system will only allow patients to be assigned to these entries. Filler entries can be created and placed between each Account Entry and the users to model the account's actual structure. Some accounts may want to see reports organized by Division, City, Hospital, Users, and Patients, while others may simply want Location, Users, and Patients. The simplest form is to enter all users directly under an account.

Access rights and privileges within the system will be associated with each entry. The root entry is special since this account has full access to the system. Some entries, such as location, may not even be allowed to

log in, as they only establish a reporting framework. Access to various applications and features within the system will be set on an entry by entry basis. Some Care Managers may be granted more privileges than others.

Programs are tied to these hierarchy entries. Programs that are tied to a user (Doctor or Care Manager) will be considered private. Whereas, programs tied to an account are considered shared. The system will support

## *Implementation Notes*

**Scaling**
Several tactics have been employed to support ten's of thousands of users and millions of appliances. An account must fit on a single Application Server. However, utilizing SuperNova, an arbitrary number of Application Servers may be established to spread the load.

By configuring the SuperNova distribution of new content and results to send copies to an aggregation server. Aggregation reports, combining data from multiple accounts, can be generated in parallel by combining data from many servers while the individual Application Servers continue to run, supporting users and appliances.

Normally, an account database can not be split across multiple servers. However, the possibility of spreading the load is possible by utilizing replication technology. Oracle is one of many database systems with automatic replication services. This permits multiple Care Managers and Doctors within an organization to each operate on a local database and stay in sync with the centralized organization server. Again, this provides a way to scale servers and ultimately grow the system to very large sizes. This is probably not low hanging fruit. The amount of effort necessary to implement this setup is not know at this time. However, this problem has already been solved in other systems.

The depth of the content hierarchy has been kept as shallow as possible to remove the likelihood of recursive queries. Wherever possible, the schema has been designed so that all related elements can be grabbed in one query and assembled or sorted within the client.

**Efficient Hierarchy / Database Interaction**
Rather than implement recursive SQL queries to grab and build hierarchies. Use the account index to grab all entities with one query. The HHN hierarchy can also be retrieved using the same method. If necessary both can be retrieved in a single query by combining (acct# eq searchacct#) or (acct# eq 'CORPHHN').

**Cube State Retrieval**
Retrieving a Patient's Cube State for a given point in time will occur very often during system processing. The most efficient means we've found of retrieving a cube uses the following mechanism:

1. Cube Metrics or Values are stored in a table indexed by Cube ID, AoC, Expression, DateEffective.
2. An additional field is added to this table called Replaced Date.
3. A Database trigger is responsible for updating the Replaced Date field whenever a new record is inserted. This places the update in the most efficient place, saving wire-time, SQL parsing, etc.
4. This enables a single query to grab the entire cube for a given patient at a given time as well as the current state by using the following queries:
   - `select * from metric where paient_id=our_patient and create_date<search_date and (isnull(replace_date) or replace_date<search_date);`

**Content Exchange**
App Server #'s must be globally unique
Account#'s
Globally Unique Object ID's
Combining Account# + CM ID, etc in BDBs allowing diff Recnums in different schemas

**Data Aggregation**
Patient GUIDs
Using Database Replication to spread load within an organization

**Program Security**
The security model in FlexCube is tied to Account Entities. Account Entities will be granted or denied access to critical system objects and applications on a per user basis. Only the Root HHN account will be able to access data across accounts. Encryption will be used to protect patient data over the Web.

**Status Group**
Status groups are a standard mechanism for tracking the status of patients, account entities, programs, and many other objects within the system. Each status group consists of Current Status, Previous Status, Transaction#, Transaction Date, and Date Effective. Each object has a private set of codes, reflecting all the states unique to that object. These are kept in the centralized STATUS_CODES table within the schema. The STATUS_HISTORY table keeps track of all changes made to each object's status.

Features:
- Current Status – Is this an active patient?
- If Current Status is the same as Previous Status, the object status has never changed
- Status History, allowing the system to know what an object's status was at any given point in the past
- Efficient synchronization of objects between Application Servers

Whenever a status change event occurs, the following database changes are triggered:
- The Current Status is copied into the Previous Status
- The Current Status is changed to the new status code
- The Transaction date is updated
- A new System Transaction # is assigned and filled in
- The System Date/Time is put into the Date Effective field
- The Database record is updated in the database
- A new STATUS_HISTORY record is created with the following info: Status, Previous Status, System Transaction #, Transaction Date, and Date Effective

## *Thoughts*

- Can we roll question node children into a variable length field within the question node? Will this present a problem when processing results data?
- Can we eliminate the Account# table and leverage off the account entity table, simplifying our overall schema?
- Account Entity – Maintains account hierarchy, includes support for
- Patient
- Group
- Dialog Editor / Composer

– Enables the creation and maintenance of non-personalized dialogs within libraries, dialog labeling, and import/export capability to the application server database

## *Issues*

| Issue | Solution |
|---|---|
| How to move Content & Results, etc. between systems with different RECNUM links. | Establish Globally Unique Ids for each object that must be moved. |
| Program Sharing | |

# Health Buddy Database (BDB)

Health Buddy Databases (BDBs) are the generic term for XML data transported between system components. BDBs are organized as miniature relational databases, enabling rich content, results, and box control information to interchangeably move between Application Servers and Appliances. Also, by utilizing standard XML, foreign systems will ultimately parse and process the information contained within HHN's systems.

Previously, survey content had to be written into script language code and compiled. BDBs remove byte-code overhead, reducing communications expense, and allowing the content to be manipulated as data, even by systems that don't support our VM. A script application can open a content BDB and present sessions in varying ways, depending on patient requirements. For example, patients with poor eyesight might be presented content in a larger font.

A generic Document Type Definition (DTD) is defined for all HHN BDBs. Specific formats for content, results, etc. are defined for each required type of data.

Initially, C code libraries will enable the data-center and appliances to parse and manipulate BDB data. Later, equivalent Java libraries will be written.

## *Appliance Requirements*

- Standard XML with BDB DTD
- Support Multiple Tables – Simple RDBMS style
- Support 32 bit Signed Integers, Date/Time stamps, and Variable length binary strings
- Date/Time stamps encoded using 32 bit Integers in Unix format
- Support Auto-Increment Integer Type
- Self-contained for transmission between systems
- Compactable for efficient transmission between systems
- Must utilize memory efficiently
- Appliance code must be small
- C & Java must be able to work with same format
- Fast storage and retrieval of field values
- Basic Schema info – Table and Field names - embedded in data
- BDB and Content Format versions
- Ability to upgrade library code, enabling shared content in a multi-threaded environment
- Consider explicit One-To-Many record linking mechanism
- Arbitrary Indexing for efficient reporting (Temp & Perm)

## *Required BDB Appliance Formats*

- Dialog
- Patient Info
- Patient Program Cube
- Session Schedule
- Raw Dialog Session Results
- Box Control
- Error Log

## *Implementation Notes*

A uniform API will be developed in both C and Java. The in-memory format for BDBs will not be the same as the transport format. However, this will be hidden behind an opaque API. All BDBs will be freely convertible between internal appliance and XML format.

**Initial API**
This is a current list of API functions/methods for the C implementation. Note: This design has not been finalized.

```
Init_BDB, Create_BDB, Destroy_bdb

open_bdb, close_bdb, debug_print_bdb

create_index, delete_index, get_index_count, select_index

add_field, get_field_count, get_field_type, get_field_name

get_record_count, new_record, modify_record, commit_record,
abort_record

first_record, last_record, next_record, prev_record

get_numeric_data, set_numeric_data

get_string_data_length, get_string_data, get_substring_data,
get_string_data_malloc, put_string_data, put_substring_data

find_record_numeric, find_record_string
```

## *Questions*

- Should we have a global DTD and separate DTDs for each object?

# Cube Object

Cube Objects are responsible for maintaining patient state throughout the life of a program. Metrics are stored within the cube and organized by Aspect of Care, Expression, and Date/Time recorded. Typical Aspects of care for a diabetes program might be Foot Care, Eye Care, Blood Glucose Management, and meter Data. Expressions include Behavior, Signs & Symptoms, and Knowledge. Additional expressions may be added to the system later on. Several different types of data are maintained within Cube objects: System and Program Variables, Refined Output Data, and Meter Readings. Cube objects manifest themselves in different ways throughout the system. They are kept in part of the application server database, travel to appliances wrapped inside a BDB, and are maintained in summary within each appliance. Internal formats for Cube objects are based on a relational database model.

## *Requirements*

- Stores metrics by Aspect of Care and expression
- Standard risk level metrics with values: Unknown, (1-3) Low Risk, (4-6) Medium Risk, and (7-9) High Risk
- Tag the Metric with a session/dialog reference so that we can trace back to the info that generated the metric
- Data Types
  - Raw Response Data linked to metrics
  - Refined Data from Output Nodes
  - Meter readings
- Implementation format must be efficient for common reporting requirements

## *Implementation Notes*

## *Issues*

# Dialog Object

Dialog objects are similar to sub-surveys that interact with patients, take meter readings, calculate metrics, and update each patient's cube state. They are kept as generic templates within a common pool, organized by library, inside an Application Server Database. Various parts of the Application Server Schema refer to dialogs in the pool, specializing them by maintaining a list of tags. Dialog objects are transported between application servers and appliances wrapped up in BDBs. All data generated by Dialogs are written to Response BDBs and Cube BDBs. A summary version of each patient's cube is maintained within the appliance, allowing access to cube and response data from within dialogs. This mechanism is the only way for dialogs to share and pass data.

Dialog objects are composed of Interaction, Evaluation, and Mapping sections. The interaction section consists of questions and branching decisions. The evaluation section contains a list of mathematical expressions that evaluate to metric values in the cube. The mapping section links each mathematical expression in the evaluation section to one or more metric addresses in the cube. Each map link specifies an Aspect of Care and Expression.
– Interacts with patients and provides output to update cube state

## *Requirements*

- Dialog nodes (Puzzle Pieces)
  - Question / Answer (Response) – Standard input methods: Short, Long, Multiple Choice, Prompt, … Note: There are two flavors of question puzzle pieces – Start Question and Followup Question. Only the Start Question is anchored on the work area.
  - Prompt – Stops and provides text info for the patient
  - Output – Writes output to a patient's cube – composed of a metric calculation section and a destination map into the patient's cube
  - Decision – Allows branching based on expressions
  - Calculation – Calculates a value using raw response, cube, and mathematical expressions
  - Goto – Jumps to a label piece
  - Label – Provides a target address for goto's
  - Stop – Halts execution of a given dialog
  - Meter – Stops execution and takes a meter reading
- Properties
  - Name – Each dialog is named and is referred by name in various parts of the system
  - Library – Each dialog belongs to a global library
  - Tags – Tags indicate special properties
    - Default Scheduling frequency
    - Default Reporting status
    - Last date/time executed
- Localization versions within Dialog Object

# Appliance

## Appliance Summary

The appliance requires additional functionality in order to support the FlexCube architecture. A To Do List architecture opens up the appliance for many-to-many communications and enables multiple users on an appliance and multiple programs or content sources per user. Content, Results, E-Commerce Transactions, User and Appliance information will all be transported as BDB XML.

### *Requirements*

- To Do list architecture
- XML / BDB content, results, e-commerce, user and appliance information
- New system applications

## Appliance Applications – To Do List Architecture

Initially, appliance code will be written in C/C++. Later, an enhanced scripting language/VM will enable the applications to be written in our scripting language. These scripts will be responsible for managing the overall box behavior. Responsibilities include Scheduling and handling appliance communications, presenting content, etc.

### *Requirements*

- Box Manager Application
- Communications Application – Overall responsibility for non-conversational communications
- Connection Application – Brings up the physical communications link
- Login Application – Logs in and establishes a secure connection
- Mail Exchange Application
- Session Application
- Presenter Application

## Box Manager Application

The Box Manager Application waits for one of two events to occur. When it's time to call into the data center, the Box Manager calls the Communications Manager and exchanges content, results, etc. Whenever a button is pushed on the appliance, it calls the Login application to initiate sessions and allow patients to initiate sessions. While it is idling, it may indicate content availability and check system integrity.

### *Requirements*

- Handles call-in schedule (Box hardwired with call-back failsafe)
- Calls Communications Manager
- Calls User Login Manager
- Calls Session Script upon user authentication
- Schedules Results BDB for delivery to application server

## User Authentication / Login Manager

- User BDB enables/disables user login requirement and style
- Styles
  - Select User from List
  - Authenticate via Luggage Lock (Clapp's method)
  - Authenticate via Smart Card
  - Authenticate via PIN

# Communications Manager

## *Requirements*

- Calls Connection Manager
- Calls Login Manager
- Calls Mail Exchange Manager
- Returns success/failure

# Connection Manager

The Connection Application will be responsible for dialing into our service provider (ISP or EDS) and establishing the physical communications link. BDBs will provide a prioritized list of local connection numbers. A fallback 800 number will dial back to HHN whenever the local connections are unavailable.

## *Requirements*

- Brings up the physical communications link
- 800 Number for Fallback
- Prioritized list of local numbers

# Login Manager

The Login Application is called after a physical link has been established. This application is responsible for logging into the host system, authenticating the box, establishing a secure connection, and navigating to the proper host service prior to data exchange. The scripts and other required information will be embedded in BDBs.

## *Requirements*

- Logs into host system
- Authenticates box
- Establishes secure connection
- Navigates prior to data exchange

# Mail Exchange Manager

## *Requirements*

- Retrieves new content and message BDBs
- Uploads new results and message BDBs
- Uses SMTP & POP3 protocols
- Will likely use non-standard ports to promote security

# Session Manager

The Session Application provides the To Do list functionality for multiple content types. Intially, FlexCube content will be the primary content.

## *Requirements*

- Calls FlexCube Presenter Application to execute content BDBs
- Prioritized with Round-Robin resolution.
- Do not present until time/date property
- Expire after time/date property
- Required Flag (This must be run before any lower priority content BDBs can execute)

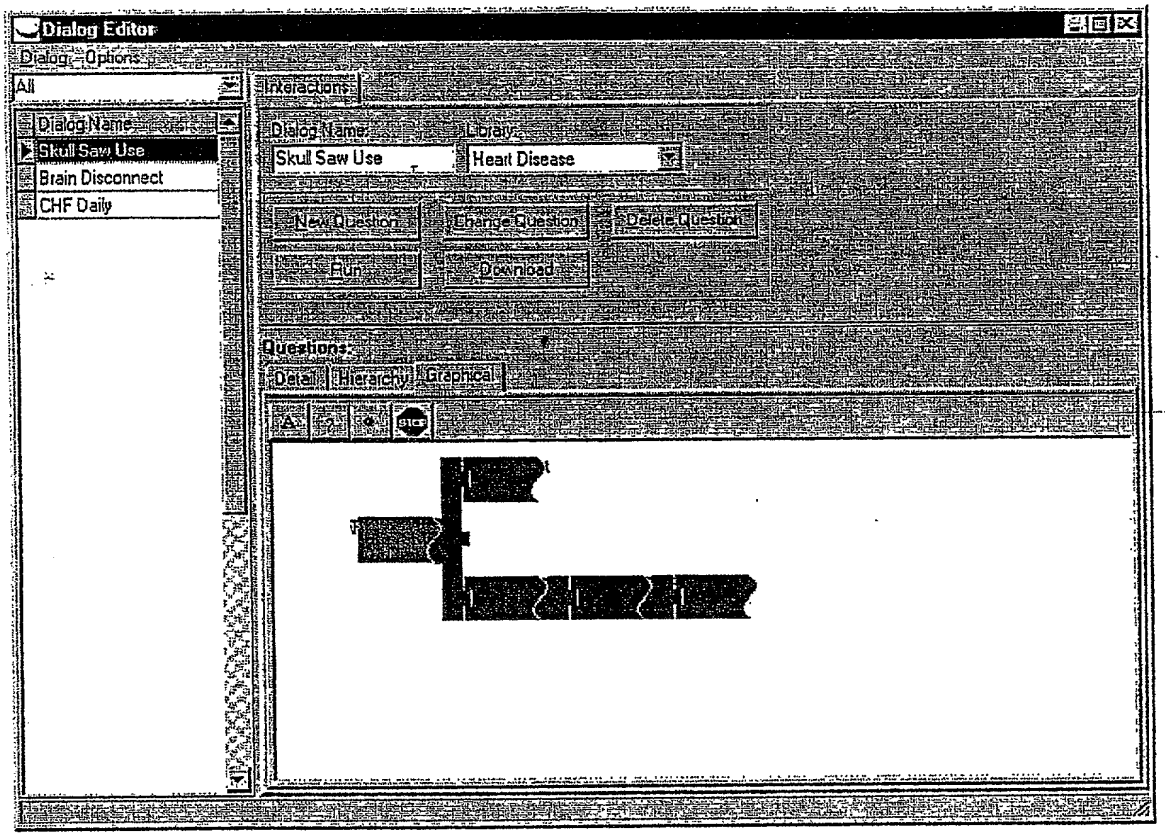
# FlexCube Presenter Application

The Presenter Application is responsible for presenting FlexCube content to the patients.

## *Requirements*

- Presents content packaged in BDBs to patients
- Writes raw question responses and evaluations to a BDB
- Updates Box cube state
- Supports standard input methods

# Windows GUI Applications




## Dialog Editor

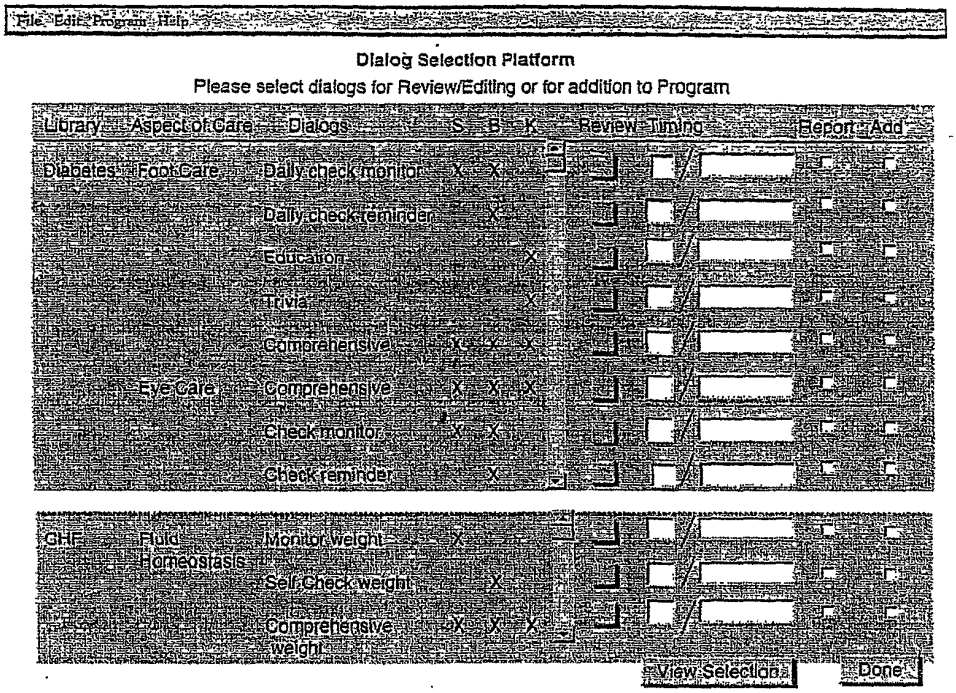
The Dialog Editor application is the primary tool for content development.

### *Requirements*

### *Requirements*

- Experience Levels
  - Beginner View – Holds author's hands, but doesn't allow full use of system
  - Expert View– Enables all features, expects author to fully understand system
- Editor Views
  - Graphical Editing view
  - Hierarchy view
  - List view
- Segregate Dialog Pool into libraries
- Select Default Dialog Properties
  - Name – Each dialog is named and is referred by name in various parts of the system
  - Library – Each dialog belongs to a global library
  - Tags – Tags indicate special properties
    - Default Scheduling frequency
    - Default Reporting status

# Program Selector

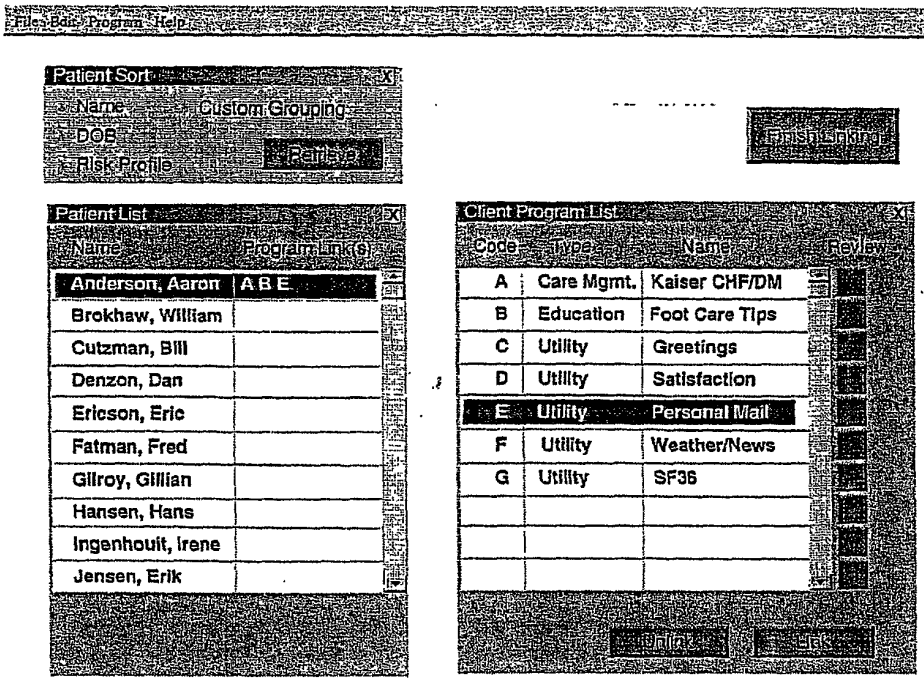



Program Selector is the application that enables Care Managers to build programs by selecting a set of dialogs from the system pool.

## *Requirements*

- View/Select available dialogs by
  - Library
  - Body System Tag
  - Name
- Set Program Properties
  - Start and End Dates
  - Inclusion and Exclusion weekdays, dates, and date ranges
  - Default Reporting Status
-

# Program Linker




Program Linker is the application that links and unlinks patients and programs. This application also enables Care Managers to set program parameters on a patient by patient basis.

## *Requirements*

- Link/Unlink an arbitrary list of patients into or out of an arbitrary list of programs
- Set patient specific program properties
  - Start / Stop dates
  - Include / Exclude specific weekdays
  - Include / Exclude specific dates and date ranges
  - Select and Approve Patient Risk Levels
- Select/Deselect patients using standard Windows GUI
  - All
  - Range
  - Individual Select/Deselect
- Sort List by
  - Name
  - Date of Birth
  - Risk Profile
  - Custom

# Session Scheduler

Session scheduler is the application responsible for scheduling and managing content presentations on the appliance. It can run as either an automatic engine or as an interactive utility. It is also useful for reviewing and editing session data.

Initially, Session Scheduler will run in semi-automatic mode. Care Managers will be able to select a patient / program combination and then view the scheduled sessions. Filter control will be provided

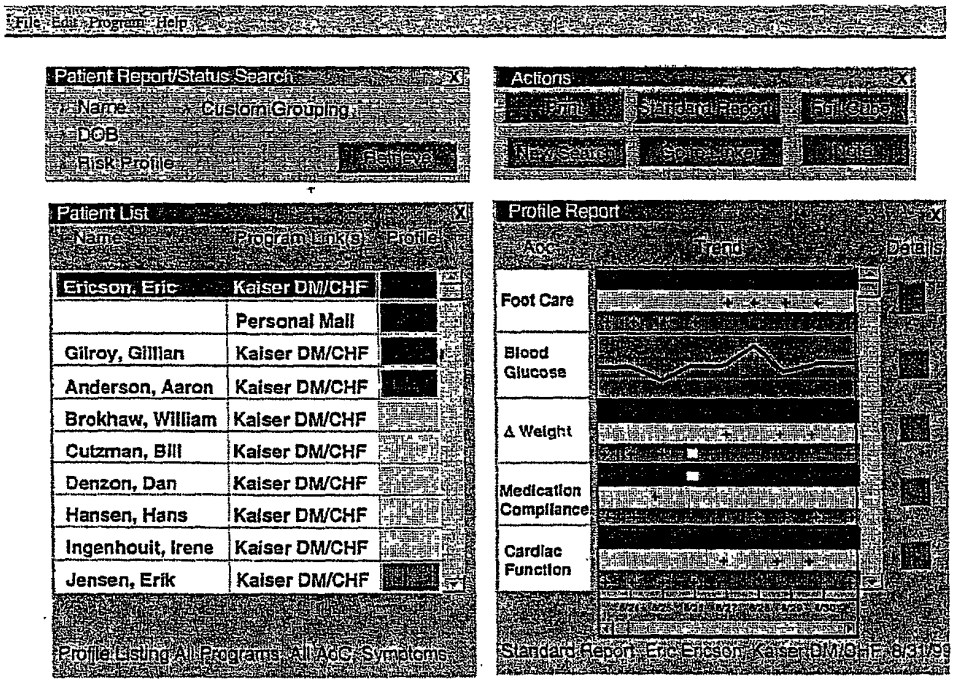
## *Requirements*

- Manual and Semi-Automatic Session generation
- Ability to view and edit listed sessions
- Session status indication – Generated, Deployed, and Completed
- Session Boundries
  - Dialogs prioritized within a session
  - Limit on total dialogs in a session
  - Dialogs selected based on individual cube state
  - Dialogs selected based on program frequency
  - Round-Robin Dialog selection for equivalent dialogs
  - Greeting dialogs with highest priority
  - Exit dialogs with lowest priority
  - Must Execute Flag – This dialog must execute without allowing patient choice before lower priority sessions are presented as to-do items
  - Inclusion / Exclusion periods for patients – blackout a vacation period
- Intelligent Session Deployment
- Initially, Care Managers will be required to approve sessions

## *Future*

- Support for Population Session creation and scheduling
- Fully Automatic Session generation

# Reporter




Reporter is the primary application for reviewing patient results and status as well as paper reports. The first suite will be written as a fat client. Later, much of Reporter will be duplicated using a thin-client HTML only model.

## *Requirements*

- High Level Patient View
- Clickable Human Body in patient detail view
- Patient Summary Report
- Population Summary Report
- Raw dialog results viewer
- Report generation summarized by hierarchy

# Administrator

Administrator is the application that manages overall system operation and configuration.

## *Requirements*

- Account Entry / Maintenance / Status Changes
- Patient Entry / Maintenance / Status Changes
- Copy Programs between Account Entities
- System Config, Patient Entry, Account Entry

# Future Additions

## Enhanced Scripting Language

The scripting language will be enhanced in several areas with the addition of structures, BDB manipulation statements, and communications operations. Structures are collections of simple variables, much like a form. Structures make it possible to manipulate data extracted from a BDB as records. Scripts will be able to open BDBs and manipulate data. New communications functions will allow scripts to communicate with other systems using the modem or serial port.

### *Requirements*

- Structures passed by value into functions
- Structures must support existing primitive types: 32bit Signed Integers & Variable length binary strings
- Full HAL functionality
- Serial & Modem communications support
- Date & Time types

## Thin-Client Applications

A production version of the system needs to supply a thin-client web view for checking patients, scheduling content, and basic system functions.

Thin Client Features

- Universal access via HTML browser to the system over the internet
- Global Access to data
- Eliminates deployment issues

BLACK LOWE & GRAHAM
816 2nd Ave., Seattle, WA 98104
206-381-3300 [HERO-1-1111]

## Questions

- In the first prototype, should we aim to create sessions on a program basis, patient basis, or both? (Patient Basis First)
- Should we build a schema capable of multiple accounts? (YES)
- Can we really merge raw data into the cube? (Probably, but it would be more efficient to store seperatly)
- Should a patient have a cube for each program?

## Thoughts / Issues

- We need to support shared puzzle-pieces
- Maybe have a Global AoC for things like HMO procedure approval screens: Your responses indicate physician attention is required, please call Dr.Smith if you do not receive a call from your doctor by tomorrow...
- Need easy way to leverage into thin-client
- Can we ask Mitch to include us in core-engineering discussions
- Wizards
- GUI hand-holding methods – Help, etc...
- Sharing of Dialogs between accounts
- Moving data between data-centers
- Dialog Interaction Precaution Tags
- Dialogs that refer to outside AoCs
- Development Hardware
- Investigate DB2

## Input

- 8/23/99 - All new patients get tossed into a special group "New Patients(5)" to make it easier for CMs to put into programs. Avoid one big pile of patients where you can't tell who is new, etc...

### *Albert Schema Questions*

- Many to many on Patient / Account Entity
- How do we move data between DBs? I'm sure this problem has been solved before in various ways. But, we need to figure out how. One thing I think we can do is integrate account # into the XML that we send. Even if it's not part of a given table. This may assist in reassembling the relationships at the destination. Another note: Account#s must be globally unique between all data centers.

### *To Purchase*

- DBA Tool(s)
- VMWare
- Order Oracle? (v 8.03/8.04 issues – CB4 Driver requires a version higher than we have...)

## System Features / Requirements

- Hidden AoCs within Cube
- Escalations

### *Escalations*

### *Minimal time and effort required to manage content*

BLACK LOWE & GRAHAM
816 2nd Ave., Seattle, WA 98104
206-381-3300 [HERO-1-1111]

# Glossary

**Application Server** – Computer system including appliance communications and thin-client web front-end and a database of content, patients, account entities, scheduling, and results.
**Aspect of Care (AoC)** – A topic within the Cube Object. Aspects of care for a diabetes program might be Foot Care, Eye Care, Blood Glucose Management, and One-Touch BG Meter.
**BDB** – A micro database object that moves cube content, dialogs, and raw response information to and from the appliance. BDBs travel in XML form.
**Database Schema** – Stores dialog and program content
**Cube Object** – Stores patient state. Includes Raw Response Data, Refined Patient Data, and Meter Data.
**Dialog Editor / Composer** – An application for creating and maintaining non-personalized dialogs within the Dialog Pool in an Application Server.
**Dialog Node** – An executable object within a dialog. These correspond to the puzzle-pieces in graphical view. Examples include: Question, Output, Goto...
**Dialog Object** – Interacts with patients and provides output to update cube state
**Dialog Pool** – The section of an Application Server that contains content dialog objects.
**Expression** – A subtopic within the Cube Object. Expressions are typically Signs & Symptoms, Behavior, and Knowledge.
**Globally Unique ID** – Unique identifiers for patients and other database objects, enabling data portability and aggregation.
**Library** – The dialog pool is partitioned into libraries. An attribute of a dialog. Typical libraries might be Diabetes, CHF, or Utilities
**Localization** – The ability of a system to support non-English languages
**Metric** – A measurement, most metrics are risk levels or meter readings. The third axis of the cube
**Personalizer** – Maintains patient-specific cube metrics, defining risk levels, etc.
**Program Selector** – Selects dialogs from libraries in and out of content specific programs, specifies labels for scheduling and reporting
**Puzzle Pieces** – A reference to the graphical objects in the dialog editor. Puzzle pieces include: Start Question, Followup Question, Prompt, Output, Decision, Calculation, Goto, Label, Stop, and Meter.
**Reporter** – Patient results lookup organized into levels from patient-specific program overview to raw results from particular session, including graphical body system view.
**Schema** – A database design, specifying the data stored and relationships between different parts of the database
**Script** – Byte code used to implement appliance applications and evaluate expressions within a dialog
**Session Application Script** – This is a compiled byte-code program that presents content to a patient. In FlexCube, this application will present dialogs and manage cube-state within the appliance as well as manage synchronization between the appliance and data center.
**Session Scheduler** – Semi-automatic creation and scheduling of patient specific sessions
**SuperNova** – An archive and distribution infrastructure for HHN system components. FlexCube is the data. SuperNova is the highway.
**Tag** – An attribute associated with a given dialog / patient or dialog / program. A typical tag might be presentation frequency.
**XML** – A text based format for encapsulating and transporting data between systems. The generic format used for HHN's data is called a Buddy Database (BDB). Acronym for Extensible Markup Language.

BLACK LOWE & GRAHAM
816 2nd Ave , Seattle, WA 98104
206-381-3300 [HERO-1-1111]

# Glossary

**Application Server** – Computer system including appliance communications and thin-client web front-end and a database of content, patients, account entities, scheduling, and results.
**Aspect of Care (AoC)** – A topic within the Cube Object. Aspects of care for a diabetes program might be Foot Care, Eye Care, Blood Glucose Management, and One-Touch BG Meter.
**BDB** – A micro database object that moves cube content, dialogs, and raw response information to and from the appliance. BDBs travel in XML form.
**Database Schema** – Stores dialog and program content
**Cube Object** – Stores patient state. Includes Raw Response Data, Refined Patient Data, and Meter Data.
**Dialog Editor / Composer** – An application for creating and maintaining non-personalized dialogs within the Dialog Pool in an Application Server.
**Dialog Node** – An executable object within a dialog. These correspond to the puzzle-pieces in graphical view. Examples include: Question, Output, Goto...
**Dialog Object** – Interacts with patients and provides output to update cube state
**Dialog Pool** – The section of an Application Server that contains content dialog objects.
**Expression** – A subtopic within the Cube Object. Expressions are typically Signs & Symptoms, Behavior, and Knowledge.
**Globally Unique ID** – Unique identifiers for patients and other database objects, enabling data portability and aggregation.
**Library** – The dialog pool is partitioned into libraries. An attribute of a dialog. Typical libraries might be Diabetes, CHF, or Utilities
**Localization** – The ability of a system to support non-English languages
**Metric** – A measurement, most metrics are risk levels or meter readings. The third axis of the cube
**Personalizer** – Maintains patient-specific cube metrics, defining risk levels, etc.
**Program Selector** – Selects dialogs from libraries in and out of content specific programs, specifies labels for scheduling and reporting
**Puzzle Pieces** – A reference to the graphical objects in the dialog editor. Puzzle pieces include: Start Question, Followup Question, Prompt, Output, Decision, Calculation, Goto, Label, Stop, and Meter.
**Reporter** – Patient results lookup organized into levels from patient-specific program overview to raw results from particular session, including graphical body system view.
**Schema** – A database design, specifying the data stored and relationships between different parts of the database
**Script** – Byte code used to implement appliance applications and evaluate expressions within a dialog
**Session Application Script** – This is a compiled byte-code program that presents content to a patient. In FlexCube, this application will present dialogs and manage cube-state within the appliance as well as manage synchronization between the appliance and data center.
**Session Scheduler** – Semi-automatic creation and scheduling of patient specific sessions
**SuperNova** – An archive and distribution infrastructure for HHN system components. FlexCube is the data. SuperNova is the highway.
**Tag** – An attribute associated with a given dialog / patient or dialog / program. A typical tag might be presentation frequency.
**XML** – A text based format for encapsulating and transporting data between systems. The generic format used for HHN's data is called a Buddy Database (BDB). Acronym for Extensible Markup Language.

BLACK LOWE & GRAHAM
816 2nd Ave., Seattle, WA 98104
206-381-3300 [HERO-1-1111]

# APPENDIX B

BLACK LOWE & GRAHAM
816 2nd Ave., Seattle, WA 98104
206-381-3300 [HERO-1-1111]

# Consumer Applications: Long Distance Care High-Level Requirements and Design

DRAFT




2505 Meridian Parkway Suite 175
Research Triangle Park NC 27713

BLACK LOWE & GRAHAM
816 2nd Ave., Seattle, WA 98104
206-381-3300 [HERO-1-1111]

# Table of Contents

# 1 Vision

The vision of the consumer applications effort is to provide applications outside the core HHN/OLS that extend the definition of the care manager and grow the HHN business into the consumer-related areas. These areas are in the health care information technology arena, but are driven by consumers and not by a hospital or HMO. Applications currently under consideration involve partnerships with:

- caregiver.com – Long Distance Care
- Stadtlanders – Care Management of Stadtlanders (medication) customers
- Weight Watchers – Management of people in Weight Watchers programs

## 1.1 Vision for Long Distance Care

The largest group of care managers in this country are family members taking care of aging parents and relatives. Family care giving has traditionally been a role of women in their 40's and 50's. Many of these women are now baby boomers in the workforce. Also, the mobility of the baby boom generation has resulted in many children living far away from their aging parents and other aging family members as well. These two trends have helped create an opportunity for technology-enabled, remote care management of elder relatives by baby boomer children.

## 1.2 Documentation

This document includes materials from the following documents that have been produced within HHN:

HHN LDC Preliminary HLRD.doc

Informal Caregiver Requirements.doc

Informal Caregiver Requirements.doc

cg pricing.xls

Term Sheet for Careguide.doc

# 2 Scope

The scope of this project is an application built around long distance care. The Health Hero Network Long Distance Care Service (HHN/LDC) is envisaged as a value-added service to the HHN/OLS that enables care providers such as family members to check on relatives in assisted living facilities. The relatives would each have a Health Buddy, and the HHN customers would access information via a thin web client.

HHN/LDC would tie into the HHN/OLS to communicate with the people in assisted living facilities who would answer daily dialogs by using the Health Buddy. The customer would create personalized content for the family member, be informed of certain alert situations, and receive reports on the family member's status.

The HHN/LDC application is in the concept stage and is currently being discussed with Careguide.com. This document represents HHN Software Products' preliminary understanding of the application based on input from Steve Brown, HHN marketing, and HHN clinical and provides a summary of high-level requirements and a high-level design for the HHN/LDC application.

## 2.1 The Market

10 million seniors live alone, and 2 million seniors live in assisted living facilities. The numbers are growing rapidly because of the aging population. While this drives the care management and assisted living businesses, it is also creating huge unmet needs for family members.

HHN has test marketed the concept of an internet based service that would allow family members to monitor parents with a Health Buddy, combining personal messages with canned senior wellness dialogues and web-based reports for the family member. The initial discussions included about a dozen working women who were in their 40s and 50s, were computer users, and were worried about aging parents. Every one expressed an interest in paying out of pocket for such a service "in a heartbeat." HHN has also introduced the idea to three major employers accounting for more than 75,000 employees. Two of these employers expressed an interest in offering such a service through their Employee Assistance Programs. The price we discussed was $500 for a one-year subscription.

More formal market analysis may be possible with Careguide.com as they have 1.3 million page views per month from an audience primarily composed of adult children of aging parents resulting in 17,000 referrals to assisted living providers.

## 2.2 Benefits to customers

The service can benefit the family member in assisted living:

- Stay home longer rather than going to assisted living
- Feel more secure, that someone cares
- Stay in contact with family member

# 2 Scope

The scope of this project is an application built around long distance care. The Health Hero Network Long Distance Care Service (HHN/LDC) is envisaged as a value-added service to the HHN/OLS that enables care providers such as family members to check on relatives in assisted living facilities. The relatives would each have a Health Buddy, and the HHN customers would access information via a thin web client.

HHN/LDC would tie into the HHN/OLS to communicate with the people in assisted living facilities who would answer daily dialogs by using the Health Buddy. The customer would create personalized content for the family member, be informed of certain alert situations, and receive reports on the family member's status.

The HHN/LDC application is in the concept stage and is currently being discussed with Careguide.com. This document represents HHN Software Products' preliminary understanding of the application based on input from Steve Brown, HHN marketing, and HHN clinical and provides a summary of high-level requirements and a high-level design for the HHN/LDC application.

## 2.1 The Market

10 million seniors live alone, and 2 million seniors live in assisted living facilities. The numbers are growing rapidly because of the aging population. While this drives the care management and assisted living businesses, it is also creating huge unmet needs for family members.

HHN has test marketed the concept of an internet based service that would allow family members to monitor parents with a Health Buddy, combining personal messages with canned senior wellness dialogues and web-based reports for the family member. The initial discussions included about a dozen working women who were in their 40s and 50s, were computer users, and were worried about aging parents. Every one expressed an interest in paying out of pocket for such a service "in a heartbeat." HHN has also introduced the idea to three major employers accounting for more than 75,000 employees. Two of these employers expressed an interest in offering such a service through their Employee Assistance Programs. The price we discussed was $500 for a one-year subscription.

More formal market analysis may be possible with Careguide.com as they have 1.3 million page views per month from an audience primarily composed of adult children of aging parents resulting in 17,000 referrals to assisted living providers.

## 2.2 Benefits to customers

The service can benefit the family member in assisted living:

- Stay home longer rather than going to assisted living
- Feel more secure, that someone cares
- Stay in contact with family member

The service can benefit the caregiver:

- Patient stays home longer saving the cost of an assisted living facility
- Provides quality assurance of services for patients already in an assisted living facility
- Reduced stress, improved quality of life for the patient and the care giver

## 2.3 Benefits to HHN

- Huge market opportunity potentially in the hundreds of thousands.
- Higher price possible
- Consumer private pay over the web means less contracting issues
- More control over network for future services such as e-commerce
- Additional market exposure for the Health Buddy concept

Putting Health Buddies into the consumer marketplace would expand the awareness of the Health Buddy to potential users of other HHN services. For example, people purchasing the HHN/LDC service might also be diabetics or have other family members who are diabetic. There are also many upselling opportunities e.g. HHN/LDC users might be interested in hearing about lightweight, warm robes and blankets that can be shipped to a loved one.

# 3 User Scenarios

## 3.1 Sign up/enrollment

Kate is a 40-year-old businesswoman and mother of three who lives in San Francisco. Her mother is 85 years old and has just moved into an assisted living facility, as she can no longer take care of her home by herself. Mrs. Novitsky takes several medications on a daily basis, is occasionally forgetful, and generally feels somewhat isolated and lonely.

Kate has used Careguide.com to find and select her mother's facility and remembers seeing something about a Health Hero service, she goes back to find out more about the service. After reading the description and seeing how the Health Buddy works she is ready to sign up for her mother.

Kate completes the registration questionnaire providing her contact information as well as demographic information about her mother. She closely reviews the Health Hero services agreement and acknowledges her acceptance of the agreement. She then proceeds to enter her credit card information to finalize her request for enrolling her mother in the Health Hero program.

Kate is now on the Health Hero program home page and sees that she can personalize the program to meet her mother's needs. She quickly moves through the step by step personalization wizard and enrolls her mother in a medication compliance and wellness program. As part of the medication program setup, she is prompted for information about her mother's medications. Kate is asked if she would like to receive a special alert via fax, e-mail, or pager if her mother's responses indicate that she is not taking her medications or is in need of a refill. Kate has no idea how critical this medication is so she clicks on a link that takes her to a detailed description of the medication. There she learns that this medication is very important, so returns to the program set up and selects to receive an alert via e-mail.

She then requests a birthday message for her mom's birthday on Feb 5th and a special "thinking of you" message for the anniversary of her father's death. Finally, she enters a personal greeting message to be delivered to her mother as part of the first set of dialogues.

Kate decides to accept the default set of reports and has now completed enrolling her mother in the Health Hero program. She calls her mother to tell her the good news.

Within moments of completing her enrollment, an e-mail is sent to Kate to welcome her to the program and confirm her mother's enrollment. A Health Buddy transaction is also generated to FedEx.

As FedEx ships the Health Buddy, another e-mail is sent to Kate to let her know that the Health Buddy is being delivered to her mother.

## 3.2 Program Delivery

Mrs. Novitsky has no problem following the Health Buddy's setup instructions and is excited to receive her first communication, which starts with the personal greeting from her daughter.

She dutifully responds to the questions for the first two days and anxiously awaits another message from her busy daughter. On the third day she indicates that she has run out of her blood pressure medication. This causes an e-mail to be immediately generated and sent to her daughter. Kate calls her mother to find out what the issue is. Mrs. Novitsky could not get a ride to the pharmacy to refill her prescription, so Kate calls the pharmacy and arranges to have the prescription delivered.

As time goes by, Kate has established a routine of logging into the Health Hero service weekly to review her mother's results, and entering new personal messages to be delivered during the week.

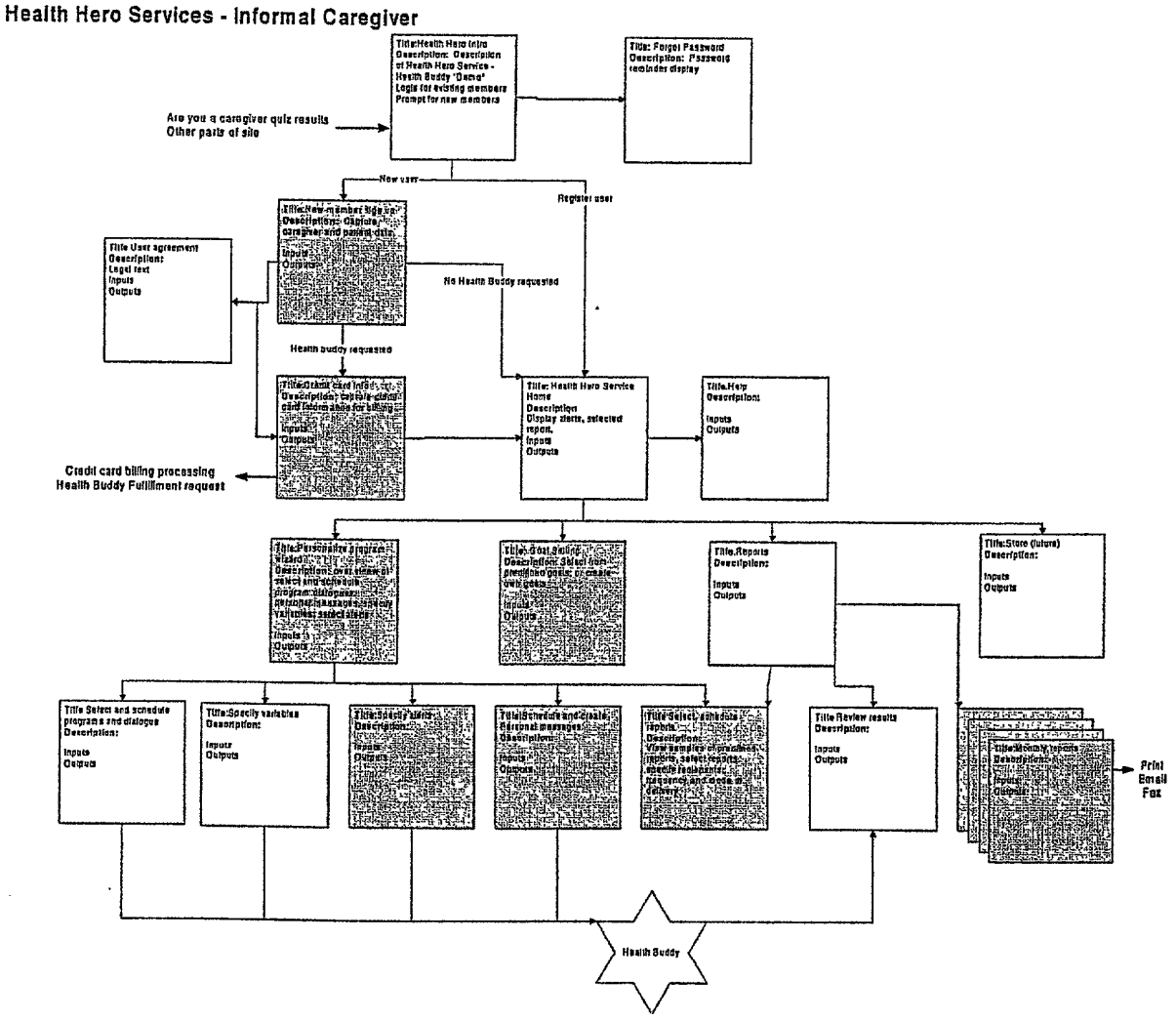
During a regular conversation with her mother, Kate learns that Mrs. Novitsky has a regular appointment with her physician coming up next week. Kate logs into the Health Hero service and requests that a copy of her mother's reports be faxed to the doctor's office for his review.

At the end of the doctor's visit, the office staff e-mails Kate that her mother is doing fine but could use some help in managing her nutrition. Kate logs into the Health Hero Service and enrolls her mother in a nutrition program, specifies her mother's ideal weight, sets an alert to occur if she loses more than five pounds in a week. Finally, she selects a link to a site that delivers groceries where she orders a regular delivery of groceries to her mom.

# 4 High-Level Requirements

This section contains high-level requirements received from marketing. These requirements currently comprise features of the envisaged service.

## 4.1 Workflow diagram



## 4.2 Health Hero Services Introduction/Splash Page

This web page will contain an introduction to the Health Hero Service and a demonstration of a Health Buddy communication. The goal is to get caregivers to sign up for the service. Specific features are:

- Prompt for user login/password
- Prompt for New User sign-up
- Forgotten password assistance
- Links to individual program descriptions
- Links to program enrollment checklists (information needed to complete enrollment for individual programs)

## 4.3 Member Sign Up

This section will allow the caregiver/purchaser to sign up and activate the HHN/LDC service. Specific features are:

**Member Registration**

- Caregiver/purchaser information – demographic, contact, login and password setup
- Health Buddy user information – demographic, contact
- User can register but not activate/request health buddy
- Sign up for a Health Buddy e-mail newsletter (generated and distributed by Health Hero)
- Send follow-up confirmation e-mail for new registrants
- Send notification letter to patient

**Activation/deactivation/upgrades/bundled medical devices**

- Allows caregiver to turn on service
- Allows caregiver to cancel service
- Allows caregiver to purchase upgrade services (i.e. BuddyLink, etc) – future requirement

**Billing information**

- Credit card information

**Health Buddy information**

- User entered information – user specific set up information
- Additional optional patient information – medications, diagnosis, physician etc.
- Health Hero maintained information (or interface from FedEx) - Health Buddy serial number, ship date (in the future this may be other device information)

- Interface to FedEx for Health Buddy Fulfillment
- Notification of shipment e-mail to caregiver from FedEx

**User agreement**

- Legal agreement/disclaimer
- Caregiver/purchaser acceptance of agreement
- Patient agreement will be achieved through a dialogue on Health Buddy

## 4.4 Credit Card Billing

This component handles the financial transaction between HHN and the purchaser of the service. Specific features are:

**Verify credit card**

- Interface to service for automatic credit card verification/validation

**Credit card billing transaction**

- Link to e-commerce server

## 4.5 Personalize Program - Wizard for Content Selection and Scheduling

This component provides a step-by-step user interface that walks the caregiver/purchaser through the process of setting up and changing programs and dialogues for a Health Buddy user. Specific features are:

**Select and schedule program and dialogues from available library**

- Select from a list of programs and dialogues defined by Health Hero staff using existing Composer functionality. Examples of preloaded content are:
  - Nutritional status (e.g. weight, appetite)
  - Medication compliance
  - Regular (e.g. monthly) assessment tools
- Enter program-specific Health Buddy user information
- Link to resources (ie Drug information, 'health handbook' etc)

**Schedule and create personal messages**

- Short message from informal caregiver to patient
- Appended to scheduled dialogue

- Message scheduled for a specific calendar day or a day of week (i.e. one time only or repeating) or to be delivered in the next communication

**Schedule and create sponsor messages/dialogues**

- Sponsor may be Careguide.com, Weight Watchers, or other third party
- Caregiver option (on behalf of patient) to receive or not receive sponsor messages
- Dialogues created by Health Hero using existing composer functionality
- Dialogues or messages assigned and scheduled by Health Hero
- Dialogues or messages appended to consumer-scheduled dialogues/messages

**Specify variables**

- Enter patient specific values for default variables previously defined in Composer by Health Hero staff

## 4.6 Personalize Reporting and Alerting Wizard

This component provides a step-by-step user interface that walks the caregiver/purchaser through the process of setting up and choosing reports and alerts. Specific features are:

**Specify alerts**

- Select alerts from predefined list. Examples might include:
  - Non-response for specifiable (e.g. 3) number of days
  - Significant (specifiable e.g. 5 lbs) change in weight
  - Request from family member for a phone call
  - Medication non-compliance (specifiable e.g. more than 3 missed doses per week)
- Specify when and how to receive alerts
  - Displayed on Health Hero Services Home Page (within Careguide.com)
  - E-mail

**Specify reports**

- Select from predefined list of reports. Examples might include:
  - Nutritional status and weight tracking report
  - Medication compliance report
  - Reports from assessment tools
- Option to preview a sample of report
- Specify report recipients
- Specify report delivery mode – fax, e-mail, display on-line

- Specify report frequency – daily, weekly, monthly, quarterly

## 4.7 Viewing Status and Reports

The caregiver/purchaser will have a personalized Health Hero Services Home Page. This will allow the purchaser to view:

- Alert messages
- Program summary
- Sponsor or Health Hero messages to caregiver/purchaser
- Links to careguide (or other sponsor) web pages/content
- Caregiver-selected reports with fax/e-mail/print/view on-line options
  - To physician
  - Other care manager
  - Others

## 4.8 Help

Help for the customer will include:

- On-line FAQ accessible via Health Hero services home page
- On-line documentation

## 4.9 Customer Service

**On-line e-mail**

- Health Hero support staff respond within 24 hours

**Telephone**

- Health Hero staff for "use of service questions"
- Call center partner for clinical/health related questions

## 4.10 Long Distance Care Program Content

The Long Distance Care program could include:

| Program Content | Personalized Content | Reports |
|---|---|---|
| Nutritional Status (weight, appetite, etc.)<br>Wellness Rating Scale<br>Medication Compliance<br>Monthly Assessment Tool -<br>• Quality of Life Tool<br>• ADL<br>• IADL<br>• Safety<br>• Depression<br>• Mini Mental Status Exam<br>Personal Hygiene<br>Skin Care<br>Nail Care (Podiatry)<br>Sleep<br>Safety<br>Nutritional Needs<br>Primary, Secondary and Tertiary Care<br>Loss and Coping<br>Social Support<br>Urinary Incontinence<br>Bowel Function<br>Activity<br>Financial Status<br>Barriers to Compliance<br>Tip of the Day<br>Trivia<br>Motivational | Personal Messaging<br>Scheduled (i.e.: Birthday)<br>Reminder Service<br>Personal Variables<br>Personal Goals | **Alerts**<br>• Non Responders<br>• Significant Change in Weight<br>• Significant Change in Wellness Rating<br>• Medication Non Compliance<br>• Request for a Call<br>• Individually Selected Key Indicators<br>**Monthly Reports**<br>• Individually Selected Variables<br>• Statement of Nutritional Status<br>• Report of Medication Compliance<br>• Monthly Assessment Tool Findings<br>• Acknowledgement and Affirmation of Status |

## Up-Selling Opportunities

- Financial Planning
- Books
- Long Term Care Facilities
- Pharmacies
- Webvan.com
- Support Groups
- Medical Support
- Durable Medical Equipment
- Insurance
- Long Term Care Planning
- Safety Products
- Drug Stores
- Card Game Sites
- Travel
- Online Shopping

# 5 High-Level Design

The currently envisaged design is a simple thin client web interface for the family member that connects to the current middleware and database for scheduling content and manages results from Health Buddies. The thin client web interface is simpler alternative to the care applets. Since there is limited composing of surveys, much of the Care Composer is not required by the purchaser/caregiver. Content will be created by HHN or the service provider and not by the purchaser/caregiver. Also, since little of the care manager functionality is required, much of the Care Director and Care Administrator is not required.

The subscriber will view everything via a web page interface, with the possibility that applets might be used in cases where more user interaction is needed, e.g. the scheduling interface. The main idea is to keep things simple for the subscriber.

The HHN/OLS v2.0 architecture currently under discussion includes migration of some functionality to a thin-client web interface and this is highly consistent with the requirements for HHN/LDC. As such, work on HHN/LDC and HHN/OLS v2.0 need to be synchronized.

## 5.1 User Interface

The user interface is determined by the actions that the subscriber needs to perform. The key actions of the family member caregiver in the HHN/LDC service are described below.

***Signing in to the LDC program.*** These screens allow entry of current users to secure locations and provide information on the program to new users that encourages them to subscribe to the LDC service.

| APPLICATION CONTENT | CONTROL | VALIDATION/RULE | DESCRIPTION |
|---|---|---|---|
| Login dialog | form | | Includes:<br>• username/password input boxes<br>• new user link – loads form to collect information on new user<br>• forgot password link – loads form with instructions for calling to get password<br>• program descriptions link – loads form with descriptions of available programs<br>• program enrollment link – loads form to capture information necessary for enrollment in individual programs |

***Signing up to the LDC program.*** These screens should include capture of data re the caregiver and the patient that enable shipping of a Health Buddy to the patient. The screens should accept the family member's credit card as payment for the program. The credit card billing infrastructure will require a link into an e-commerce server.

| APPLICATION CONTENT | CONTROL | VALIDATION/RULE | DESCRIPTION |
|---|---|---|---|
| Caregiver/ purchaser information | Form | Confirmation e-mail is sent to new caregivers. | The caregivers demographic, contact, login and password setup information is captured. Caregivers can optionally subscribe to newsletter. |
| Health Buddy user information | Form | Notification letter is sent to patient. | The Health Buddy users demographic and contact information is captured. Optional information e.g. pt medications, physician, is captured. |
| Health Buddy User Agreement | Health Buddy dialogue | | Legal agreement obtained via Health Buddy dialogue. |
| Activation | Form | | Allows caregiver to turn on, cancel or upgrade service. |
| User Agreement | Form | Caregiver acceptance of agreement required to move on. | Legal agreement, disclaimer is loaded. |
| Billing | | | Credit card information captured. Verification and validation achieved via interface to service. Transaction logged via eCommerce server. |
| Fulfillment | Form | FedEx EDI interaction completed and e-mailed from FedEx to caregiver generated. | Captures information necessary for FedEx EDI transactions. |

***Choosing LDC preloaded program, creating personalized messages and scheduling preloaded and personalized dialogues.*** These screens should allow the family member to:

- Select an LDC program and dialogues based on content that are available in the (content) database.
- Create a message (prompt) that can be scheduled to play on the Health Buddy on a specific date. These personalized messages can be stored using the patient variables structure.
- Schedule selected content. This should look like a calendar that is populated based on what LDC preloaded program is chosen and what personalized messages are created. The calendar should be editable. The scheduler/calendar may need to be an applet in order to offer the required functionality. Scheduling using the current architecture is a major issue. Please see the section describing some of the issues.

| APPLICATION CONTENT | CONTROL | VALIDATION/RULE | DESCRIPTION |
|---|---|---|---|
| LDC Program | Form | | Select LDC program(s) e.g.<br>• Nutritional status<br>• Medication compliance<br>• Regular assessment tools |

| APPLICATION CONTENT | CONTROL | VALIDATION/RULE | DESCRIPTION |
|---|---|---|---|
| LDC Messages | Form | | Create messages for scheduling with LDC program dialogues |
| Schedule Calendar | Form | | Schedule LDC program content with appended, personalized messages for delivery via Health Buddy. Can be edited. |
| Resources | Form | | Links to available Health Care Resources i.e. Drug information, 'health handbook', etc. |
| Sponsor Messages | Form | | Caregiver can elect to not have sponsor messages show up to patient *(see sponsor below) |
| Variables | Form | | Allows caregiver to enter pt-specific values instead of defaults set by HHN. |

***Choosing, scheduling, and viewing reports.*** These screens will provide a web interface for a report menu where previously created reports can be viewed, reports can be run, and reports can be scheduled to run at defined times. The reporting will require a report server on the back end as an additional piece of architecture.

| APPLICATION CONTENT | CONTROL | VALIDATION/RULE | DESCRIPTION |
|---|---|---|---|
| Alert Select | Form | | Select Alerts from pre-defined list. Specify how to receive alerts: e-mail, web. |
| Report Select | Form | | Select Reports from pre-defined list. Option to preview reports. |
| Report Delivery | Form | | Specify report recipients, delivery mode and frequency. |
| Alert and Report Viewing | Form | | Viewing options include<br>• Alerts<br>• Program summary<br>• Sponsor or Health Hero messages to caregiver<br>• Links to careguide (or other sponsor) web pages/content<br>• Caregiver-selected reports with fax/e-mail/print/view on-line options |

****Creating/scheduling sponsor messages/dialogues.*** These screens should allow for creation and scheduling of sponsor messages. This may be a screen in a separate wizard that can be run by the sponsor (i.e. Careguide.com or other third party), or created by Health Hero using existing composer functionality.

| APPLICATION CONTENT | CONTROL | VALIDATION/RULE | DESCRIPTION |
|---|---|---|---|
| LDC Program | Form | | Select LDC program(s) e.g.<br>• Nutritional status<br>• Medication compliance<br>• Regular assessment tools |
| LDC Messages | Form | | Create sponsor messages for scheduling with LDC program dialogues |

| APPLICATION CONTENT | CONTROL | VALIDATION/RULE | DESCRIPTION |
|---|---|---|---|
| Schedule Calendar | Form | | Schedule LDC program content with appended, sponsor messages for delivery via Health Buddy. Can be edited. |

The design of the user interface will be wizard-like, stepping the caregiver through the above actions. The interface will be web based so that no installation will be required on the client machines.

The user interface will be entirely new. Some of the current screens could be used as models, but most of the interaction that the subscriber will have with the system will be at a simpler level than with the current care managers. The user interface will be simple and easy for a subscriber to walk through and find what they need.

## 5.2 Server-side changes: Database and Middleware

Based on the requirements for the interface, most of the server-side changes will be in scheduling flexibility and the addition of servers/middleware to handle reporting, alerts, and e-commerce.

The key database and middleware actions in the HHN/LDC service are:

- Construction of surveys from preloaded and personalized dialogues; ready for download to Health Buddy with specific dialogues or on specific dates.
- Handling situation where survey is not answered, i.e. queuing of surveys while allowing for personalized dialogues to run in the box on a given day.
- Modifying surveys that have been constructed and are scheduled, to add further personalized dialogues.
- Scheduling constructed surveys to run on Health Buddy.
- Scheduling reports
- Running reports and returning results to files that are linked to family member web pages.
- Including graphs in reports.
- Handling of role-based security, archiving and distribution of reports.
- Handling alerts.

This database and middleware functionality requires additional middleware to be written for database communication and communication with an additional report server, graphics server and web server. Middleware changes need to be made to allow extra flexibility in scheduling. (Please see scheduling section later).

When reporting from the server, the middleware components interact with the database, the graphics server, the web server, the report server, and the report writer. There are two distinct sets of middleware: those interacting with the web server and those interacting with

the report server to create reports. The components used by the web server assist in creating dynamic web pages for display to the user. The components used by the report server handle the interactions between the graphics server, the database, and the report writer to bring all the pieces together into a completed report.

The middleware components interact with the report writer to construct the reports from database data and from graphical and textual output produced by the graphics server. The middleware controls the creation of ad-hoc reports with help from the report writer's functions. Data may be sent from the middleware or from the database to the graphics server. The middleware connects to the various databases that are supported in the application.

Changes to the current database should be fairly limited. There will be multiple additions however, since the database will have to store additional information, such as credit card data and alert information. A more in-depth review of the requirements will be needed in order to list all of the database additions and changes.

The current middleware can be used with additions and modifications to handle the additional data and the additional server-side components that are required for reporting and e-commerce.

## 5.3 Application Content

In order for this application to be relevant to a wide patient population, a significant quantity of high-quality content needs to be developed and integrated into the application. Content areas may include:

- Personal hygiene, skin care, nail care, sleep, safety, social support, urinary incontinence, bowel function, activity, financial status, barriers.
- Nutritional status
- Medication compliance

Other key tasks in the content area are:

- Content needs to be arranged to be readily scheduled into marketable programs e.g. monthly, quarterly etc.
- Content needs to be tagged to enable reporting of results.

The pre-defined application content will be created by HHN or the service provider using the Care Composer. The content will be stored, created, and reviewed using the Care Composer. The subscribers might be able to review the content but they will not be able to modify the pre-defined content. Personalized messages can be added to the pre-defined content.

## 5.4 Survey definition and scheduling

The current dialogue scheduling architecture has two significant problems in the context of HHN/LDC and other applications (e.g. HHN/DMS -Diabetes).

- There is no way to merge content at schedule time
- Only one survey per day can be sent to the Health Buddy

For the HHN/DMS system, we add personalized questions (goals) to pre-loaded surveys (education subtopics) by specifying goals as program metrics. We specify one metric that sets an upper limit on the number of goals allowed and individual metrics for each possible goal. Questions based on these metrics can then be added to each pre-loaded survey (programmatically). These questions act as placeholders for goal questions that may be asked at some point in time.

The architecture described above will work for the HHN/LDC application only if we ensure that some pre-loaded content is sent to the Health Buddy each day. Even then, if questions are not answered on a given day, surveys scheduled for the next day will not be sent on that day. The current architecture thus does not ensure that personalized content scheduled for a specific day will actually run. The best we can do in the current architecture is to allow personalized messages to be appended to dialogues that are part of the program, and for the program dialogue and message appendix to be run in sequence. However there would be no guarantee that a personalized message would appear on a particular day of the year.

Going forward, a more flexible architecture is desired. Features and requirements should include:

- Ability to combine surveys for scheduling on a given day.
- Ability to schedule surveys like appointments e.g. weekly, monthly etc. via a calendar.

The scheduling of personalized content is one area where a lot of work will need to be done to give the flexibility requested in the requirements. There are some workarounds as described above, but the flexibility desired in the requirements would not be completely attainable with the current scheduling and addition of content mechanisms. Note that all additional work on content scheduling would need to be done by, or in conjunction with, the HHN Mountain View Engineering team.

## 5.5 Reporting

The reporting component is a crucial part of the HHN/LDC. Subscribers need to easily schedule and view reports. The current system does not handle the reporting architecture that will be needed for this system.

The simplest, cleanest approach to reporting is to set up a new report server (with report writer), statistics/graphics server, and a web server. The report server will require middleware for connectivity to the transactional database at EDS and for connectivity with the report writer, statistics/graphics server, and the web server.

Given the current HHN internal processes it would be most efficient for this project (and for the v2.0 core offering) to set up separate boxes for a report server (with report writer), statistics/graphics server, and web server. These boxes would be set up and managed internally at HHN and communicate with the EDS data center only for pulling data from the transactional database as required.

The HHN/OLS v2.0 core reporting framework currently under discussion is highly consistent with the reporting requirements for HHN/LDC. As such, work on HHN/LDC and HHN/OLS v2.0 needs to be synchronized. In particular, requirements and design for the reporting component of HHN/OLS v2.0 need to be defined before any work on the reporting component of HHN/LDC is done. HHN Software Products has a Preliminary High-Level Requirements and Design document for HHN/OLS v2.0 Batch Reporting. That document is currently in circulation within HHN Engineering.

Since the reports will consist of text and graphs, there needs to be some way to combine these items and also produce the graphs without a user interface interaction. For this and other reasons, it is suggested that the reports be produced in PDF. PDF format is widespread and makes it easier to control layout than with HTML and other formats. With PDF, the report will be read-only and accessible from a variety of systems.

The following section describes a reporting architecture that provides significant scalability and workflow efficiencies. The architecture will allow subscribers to automate report generation and distribution. The following sections are consistent with the Preliminary High-Level Requirements and Design document for HHN/OLS v2.0 Batch Reporting, and the reader is referred to that document for additional detail.

### *5.5.1 Reporting Architecture*

The proposed reporting architecture is based around a thin client web-browser user interface. The user interface displays a list of reports and enables reports to be scheduled or interactively run. Report results are automatically archived and managed in a hive, are available for viewing via role-based security, and can be automatically sent to other people as appropriate.

The server-side architecture is asynchronous whereby a user connects, requests a report, and can either wait for the report or disconnect and get the report later. The asynchronous design provides more scalability since reports can be queued and

processed by the report server. The asynchronous architecture also handles problems that could occur with someone on a bad connection; if the connection gets dropped, the report will be waiting when they get reconnected. The report server allows for the scheduling of reports, if those reports are run on a consistent basis.

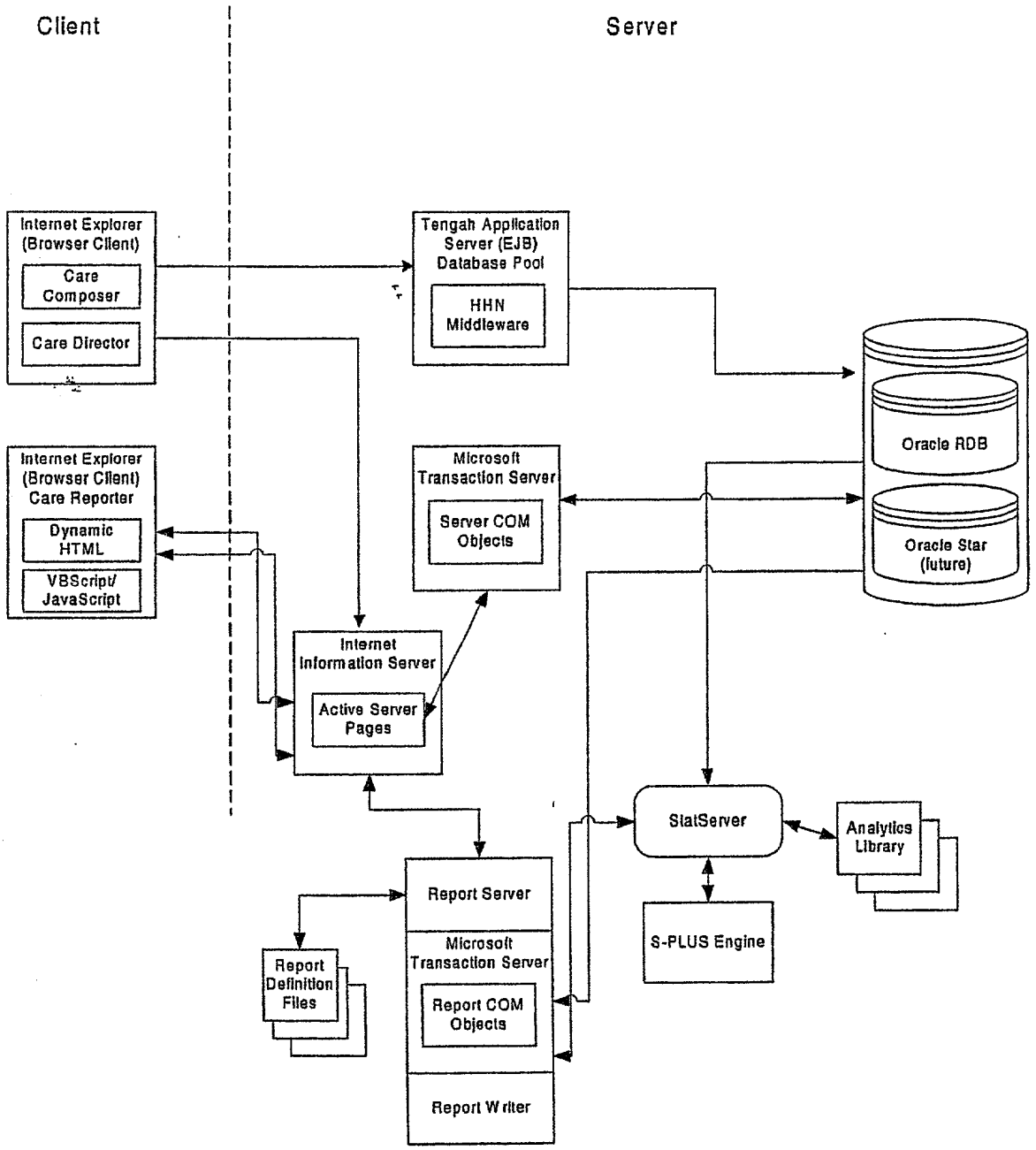
The central points of action in the architecture are the **middleware** components, which act as traffic cops handling interactions between the other components. The middleware components use the report writer to create reports that are sent back to a web page. The middleware handles getting data and analyses from the statistical/graphical engine and publishing the data or sending the output to the report writer.

The **report writer** allows for creation of report templates that can be easily modified at runtime to handle the ad-hoc nature of report options. The report writer handles the creation of the reports and the export of reports in various file formats. On the server-side, the report server uses the middleware and the report writer to process reports that are returned to the **web server**. When running reports on the server, the web server handles all user interaction with the server.

A **report server** is needed to manage the security, scheduling, and distribution of reports. There are available products that suit the needs of HHN applications without HHN having to build a report server. Currently, Report2Web has been reviewed. Others currently being reviewed are Brio.Report, Actuate, and WebFocus. The main purpose of the report server is to manage reports so that users can request reports, retrieve them later, and rerun reports using saved settings. On the web server, a user's security level determines the reports a user can run. The report server allows a user access to only the reports that the user ran or ones in "public" folders. The report server processes reports requested by users via the web server. The parameters and other report information are read from the web server and used to create the desired report with the correct options. The report server interacts with the middleware components to create reports using the report writer components and output from the statistical/graphical engine.

The **statistics/graphics server** manages sessions in which specific graphical analyses are run depending on the required reports and personal data. Currently, StatServer from MathSoft has been evaluated. This readily handles the data analysis sections of reports, creating graphs or data sets for display in reports. The **database** system is currently an Oracle database at EDS.

When the user selects a report, chooses options, and submits the report, the middleware handles the interaction with the statistics/graphics server. For example, an instance of the statistics/graphics engine is created via its Open method, and a new analysis object is created. Then, arguments and data associated with the analytic used are passed to the analysis. After this, the Run method for this analysis object is called. The results are returned to a file, which contains the information relevant to the report. The image and/or summary statistics is then embedded into the report by the middleware.


Client
Server
Internet Explorer (Browser Client)
Care Composer
Care Director
Tengah Application Server (EJB) Database Pool
HHN Middleware
Oracle RDB
Oracle Star (future)
Internet Explorer (Browser Client) Care Reporter
Dynamic HTML
VBScript/ JavaScript
Microsoft Transaction Server
Server COM Objects
Internet Information Server
Active Server Pages
StatServer
Analytics Library
S-PLUS Engine
Report Server
Microsoft Transaction Server
Report COM Objects
Report Writer
Report Definition Files

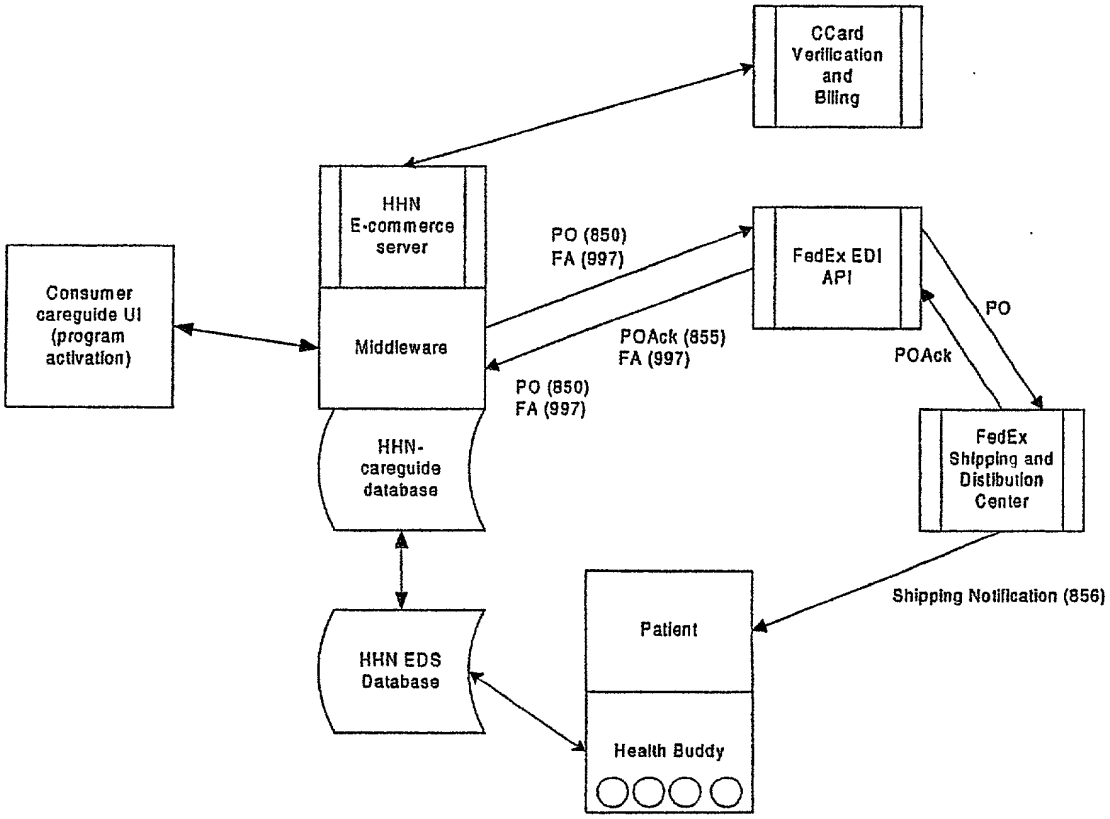
## 5.6 E-Commerce and Fulfillment

Additional server-side processing will be required to validate the credit card for the billing processes. The server will need to connect to a credit validation service to be able to process the order on-line. This will require extra middleware and/or an E-Commerce server to handle all of this processing.

A summary of E-commerce servers is provided as an appendix. For the HHN/LDC application, at first blush, we think the IBM Net.Commerce product would work well. This would need to be investigated further in association with current and anticipated future commerce requirements.

Fulfillment will involve interaction with the FedEx system currently in development. As we understand it, an API to the FedEx EDI factory will be available and we will be able to programmatically configure transactions and exchange transaction components.

The FedEx EDI interaction may look like the following:




The HHN/LDC application development will need to stay close to developments in the FedEx fulfillment system and to integrate accordingly.

## 5.7 Overall Design Issues

It is envisaged that HHN/LDC would address the requirements described above through a new thin client web user interface that facilitates communication with the Health Buddy via the HHN/OLS. The selected content topics would be linked to dialogues in the Data Center. These would then be run remotely over the Health Buddy appliance. This thin client user interface would also communicate with a report server, graphics server, and web server that would pull data from the HHN/OLS Data Center, run reports, and link reports to the family member's web page.

Many of the architectural issues discussed herein are currently being discussed by the HHN Architecture team. Some of these issues, while specific to the LDC project, apply more generally to a more flexible next generation HHN/OLS system. In particular, the architecture described above would provide a flexible, asynchronous, server-side reporting system for the core system. The scalability and the asynchronous nature of the proposed reporting architecture will provide a fast user experience while managing demands on server resources. The major areas that need improvement for this project are more dynamic scheduling of content and reporting/alerting of users.

BLACK LOWE & GRAHAM
816 2nd Ave , Seattle, WA 98104
206-381-3300 [HERO-1-1111]

# 6 Going Forward

## 6.1 Key Business Analysis Considerations

**Careguide.com**

The case for developing this product lies in the potential business relationship with Careguide.com. Specifically, given that this is a serious opportunity, an HHN team needs to meet with Careguide.com in order to gain an understanding of:

- Careguide.com's current product and product development milestones
- Key goals and requirements of this application from Careguide.com's perspective
- How and where the project will provide profits to HHN and to Careguide.com
- Risks for Careguide.com and HHN
- Who will own and develop ancillary business opportunities

## 6.2 Software Lifecycle Development

HHN Software Products has the following phased software lifecycle development process:

1. High level requirements and design
2. Detailed design
3. Development stage 1
4. Deployment as beta
5. Development stage 2
6. Commercial deployment

Typically phase 1 takes one to two months, depending on availability of solid requirements and phase 2 takes approximately two months. Milestones and schedules for stages beyond this can not be determined until the detailed design is in place.

The development of the HHN/LDC system will have to closely follow the other changes being made in the core HHN system. Many of the major changes required to make the HHN/LDC successful are in discussion as changes to the core system. The HHN/LDC requirements will be helpful in defining changes that need to be made in the core system, changes that will assist in the development of the HHN/LDC service going forward.

## 6.3 Next Steps

BLACK LOWE & GRAHAM
816 2nd Ave., Seattle, WA 98104
206-381-3300 [HERO-1-1111]

Discussions with careguide.com need to be contained to developing an understanding of the potential relationship, their current product and product development milestones, and key goals and requirements of this application from careguide.com's perspective.

Conditional on the business case and interest, the following internal resources and activities are required:

1. **HHN Software Products involvement.** HHN Software Products to meet with careguide.com in order to begin a software business analysis and software requirements/design process. Some issues involved in this process are addressed above. This work would result in a Detailed Design document for the software build.

2. **HHN Marketing involvement.** HHN Marketing to develop a concept document and a marketing requirements document for this project.

3. **HHN Clinical involvement.** HHN Clinical Department to contribute to the concept document, the MRD, and the Detailed Design document.

4. **HHN Engineering involvement.** HHN Engineering to be involved in any changes to the current middleware and Care applet functionality as described above.

5. **HHN QA involvement.** HHN QA to be involved in testing and deploying any system built, and estimates of their time would need to be included in any project plan.

6. **HHN Supply Chain involvement.** HHN Supply Chain to be involved in setting up the eCommerce server and the fulfillment and billing processes.

# Appendix: E-Commerce Servers

E-Commerce servers come in four broad classes:

**Web Storefronts** are shrink-wrapped catalog products that enable simple order entry via a web browser. They typically provide store creation wizards, sample storefronts, and a payment system. Additional functions include catalog builders, search engines, shopping carts, and order tracking. No gateways or APIs to legacy, back-fulfillment, customer support, or inventory systems are usually provided. Links are limited to EDI, messaging, screen-scraping, and flat data. Web storefronts typically run on NT and some UNIX platforms and are priced in the range $1000 to $20,000. Vendors include IBM Net.Commerce (www-4.ibm.com/software/commerce/net.commerce), Open Market (www.openmarket.com/products), SpaceWorks (www.spaceworks.com).

**Integrated Web Catalogs** are shrink-wrapped customizable storefronts with shopping carts, customer-specific pricing abilities, customer account management functions, and product sales reporting. Additional functionality includes message based or screen scraping links to inventory and accounting systems. Integrated web catalogs typically run on NT and UNIX platforms and are priced in the range $10,000 to $50,000. Vendors include CommerceOne (www.commerceone.com), Clarus (www.claruscorp.com), and Ariba (www.ariba.com/corp/AribaSolutions/overview.asp).

**Web Server Tool Sets** are components and programming tools that enable users and integrators to build EC functional servers with gateways or APIs to back-office applications. Features include shopping carts, customer-specific product configuration, customer-specific pricing abilities, and customer account management functions. Web server tool sets typically run on NT and UNIX platforms and are priced in the range $5,000 to $250,000. Vendors include Harbinger (www.harbinger.com), Microsoft Site Server, Oracle, and Procurenet (www.procurenet.com).

**Enterprise Web Servers** are non shrink-wrapped, customized solutions incorporating storefront, integrated catalog, and server tools with robust, object-oriented APIs and gateways into multiple legacy EC systems. Enterprise web servers typically run on NT and UNIX platforms and are priced in the range $50,000 to $500,000. Vendors include Connect (www.connectinc.com/) and Broadvision (www.broadvision.com).